IN THE CLAIMS:

1. **(Currently Amended)** A method of servicing a request for a document over a computer network, comprising the steps of:

receiving a request for a document, the **received** document including a script that defines plurality of blocks, each block including a reference to a data source and code that is adapted to access the data source and to format the data accessed from the data source;

retrieving at least one **but not all** of the plurality of blocks defined in the script of the requested document from a memory**, the memory storing** ~~**when the memory stores**~~ the at least one of the plurality of blocks defined in the script of the requested document; and

dynamically generating **remaining ones of the plurality of blocks** ~~**any block**~~ defined in the script of the requested document that ~~**is**~~ **are** not stored in the memory and storing a copy of each dynamically generated block in the memory.

2. **(Original)** The method of claim 1, further comprising the step of assembling the requested document from at least one of the retrieved and dynamically generated blocks.

3. **(Original)** The method of claim 2, further comprising the step of sending the assembled document over the computer network to an originator of the request.

4. **(Original)** The method of claim 2, further comprising the step of sending at least one of the retrieved and dynamically generated blocks over the computer network to an originator of the request.

5. **(Original)** The method of claim 1, wherein the document includes an XML document.

6. **(Original)** The method of claim 5, wherein the document includes an HTML document.

7. **(Original)** The method of claim 1, wherein the request includes an HTTP request.

8. **(Original)** The method of claim 1, wherein the memory is a cache memory.

9. **(Original)** The method of claim 1, wherein the memory is adapted to be shared among multiple processes.

10. **(Original)** The method of claim 1, further including the step of determining whether the at least one of the plurality of stored blocks has been invalidated and carrying out the retrieving step only when the at least one of the plurality of stored blocks has not been invalidated.

11. **(Original)** The method of claim 1, further comprising the step of determining at least one of an invalidation mechanism and an expiration time for each dynamically generated block that is stored in the memory.

12. **(Original)** The method of claim 1, further comprising the step of storing a placeholder block configured to enable an external data source to asynchronously publish data thereto.

13. **(Original)** The method of claim 12, wherein the placeholder block is free of code to access and format data.

14. **(Original)** The method of claim 1, further comprising the step of accepting asynchronous input from an external data source, the asynchronous input updating at least one block stored in the memory.

15. **(Original)** The method of claim 1, wherein the memory is maintained across a plurality of cache servers, and wherein a coherency mechanism maintains coherency of the memory across the plurality of cache servers.

16. **(Original)** The method of claim 15, wherein the plurality of cache servers are distributed over a geographical area.

17. **(Original)** The method of claim 1, further comprising the steps of associating at least one caching property to each dynamically generated block, the at least one caching property determining when the associated block is invalidated.

18. **(Original)** The method of claim 17, wherein the at least one caching property is stored along with the copy of each dynamically generated block stored in the memory.

19. **(Currently Amended)** A computer system for servicing a request for a document over a computer network, comprising:

at least one processor;

at least one data storage device;

a plurality of processes spawned by said at least one processor, the processes including processing logic for:

receiving a request for a document, the **received** document including a script that defines plurality of blocks, each block including a reference to a data source and code that is adapted to access the data source and to format the data accessed from the data source;

retrieving at least one **but not all** of the plurality of blocks defined in the script of the requested document from a memory**, the memory storing** ~~**when the memory stores**~~ the at least one of the plurality of blocks defined in the script of the requested document; and

dynamically generating **remaining ones of the plurality of blocks** ~~**any block**~~ defined in the script of the requested document that **are** ~~**is**~~ not stored in the memory and storing a copy of each dynamically generated block in the memory.

20. **(Original)** The computer system of claim 19, further including processing logic for assembling the requested document from at least one of the retrieved and dynamically generated blocks.

21. **(Original)** The computer system of claim 20, further comprising processing logic for sending the assembled document over the computer network to an originator of the request.

22. **(Original)** The computer system of claim 20, further comprising processing logic for sending at least one of the retrieved and dynamically generated blocks over the computer network to an originator of the request.

23. **(Original)** The computer system of claim 19, wherein the document includes an XML document.

24. **(Original)** The computer system of claim 23, wherein the document includes an HTML document.

25. **(Original)** The computer system of claim 19, wherein the request includes an HTTP request.

26. **(Original)** The computer system of claim 19, wherein the memory is a cache memory.

27. **(Original)** The computer system of claim 19, wherein the memory is adapted to be shared among multiple processes.

28. **(Original)** The computer system of claim 19, further including processing logic for determining whether the at least one of the plurality of stored blocks has been invalidated and for carrying out the retrieving step only when the at least one of the plurality of stored blocks has not been invalidated.

29. **(Original)** The computer system of claim 19, further comprising processing logic for determining at least one of an invalidation mechanism and an expiration time for each dynamically generated block that is stored in the memory.

30. **(Original)** The computer system of claim 19, further comprising processing logic for storing a placeholder block configured to enable an external data source to asynchronously publish data thereto.

31. **(Original)** The computer system of claim 30, wherein the placeholder block is free of code to access and format data.

32. **(Original)** The computer system of claim 19, further comprising processing logic for accepting asynchronous input from an external data source, the asynchronous input updating at least one block stored in the memory.

33. **(Original)** The computer system of claim 19, wherein the memory is maintained across a plurality of cache servers, and wherein a coherency mechanism maintains coherency of the memory across the plurality of cache servers.

34. **(Original)** The computer system of claim 33, wherein the plurality of cache servers are distributed over a geographical area.

35. **(Original)** The computer system of claim 19, further comprising processing logic for associating at least one caching property to each dynamically generated block, the at least one caching property determining when the associated block is invalidated.

36. **(Original)** The computer system of claim 35, wherein the at least one caching property is stored along with the copy of each dynamically generated block stored in the memory.

37. **(Currently Amended)** A machine-readable medium having data stored thereon representing sequences of instructions which, when executed by computing device, causes said computing device to service a request for a document over a computer network, by performing the steps of:

receiving a request for a document, the **received** document including a script that defines plurality of blocks, each block including a reference to a data source and code that is adapted to access the data source and to format the data accessed from the data source;

retrieving at least one **but not all** of the plurality of blocks defined in the script of the requested document from a memory**, the memory storing** ~~**when the memory stores**~~ the at least one of the plurality of blocks defined in the script of the requested document; and

dynamically generating **remaining ones of the plurality of blocks** ~~**any block**~~ defined in the script of the requested document that **are** ~~**is**~~ not stored in the memory and storing a copy of each dynamically generated block in the memory.

38. **(Original)** The medium of claim 37, further comprising the step of assembling the requested document from at least one of the retrieved and dynamically generated blocks.

39. **(Original)** The medium of claim 38, further comprising the step of sending the assembled document over the computer network to an originator of the request.

40. **(Original)** The medium of claim 38, further comprising the step of sending at least one of the retrieved and dynamically generated blocks over the computer network to an originator of the request.

41. **(Original)** The medium of claim 37, wherein the document includes an XML document.

42. **(Original)** The medium of claim 41, wherein the document includes an HTML document.

43. **(Original)** The medium of claim 37, wherein the request includes an HTTP request.

44. **(Original)** The medium of claim 37, wherein the memory is a cache memory.

45. **(Original)** The medium of claim 37, wherein the memory is adapted to be shared among multiple processes.

46. **(Original)** The medium of claim 37, further including the step of determining whether the at least one of the plurality of stored blocks has been invalidated and carrying out the retrieving step only when the at least one of the plurality of stored blocks has not been invalidated.

47. **(Original)** The medium of claim 37, further comprising the step of determining at least one of an invalidation mechanism and an expiration time for each dynamically generated block that is stored in the memory.

48. **(Original)** The medium of claim 37, further comprising the step of storing a placeholder block configured to enable an external data source to asynchronously publish data thereto.

49. **(Original)** The medium of claim 48, wherein the placeholder block is free of code to access and format data.

50. **(Original)** The medium of claim 37, further comprising the step of accepting asynchronous input from an external data source, the asynchronous input updating at least one block stored in the memory.

51. **(Original)** The medium of claim 37, wherein the memory is maintained across a plurality of cache servers, and wherein a coherency mechanism maintains coherency of the memory across the plurality of cache servers.

52. **(Original)** The medium of claim 51, wherein the plurality of cache servers are distributed over a geographical area.

53. **(Original)** The medium of claim 37, further comprising the steps of associating at least one caching property to each dynamically generated block, the at least one caching property determining when the associated block is invalidated.

54. **(Original)** The medium of claim 53, wherein the at least one caching property is stored along with the copy of each dynamically generated block stored in the memory.

55. **(Original)** A method of servicing a request for a Web page over a computer network, comprising the steps of:

identifying constituent blocks of the Web page, each of the constituent blocks including a portion of the Web page;

defining a caching property for each identified block, the caching property defining when each identified block is to be invalidated;

caching the identified blocks in a memory;

maintaining each of the cached blocks in the memory according to the defined caching property defined for each block, and

servicing the request for the Web page at least partially from the cached blocks in memory.

56. **(Original)** The method of claim 55, wherein each of the constituent blocks includes a reference to a data source and code that is adapted to access the data source and to format the data accessed from the data source.

57. **(Original)** The method of claim 55, wherein the servicing step includes a step of assembling the requested Web page from at least one of the cached blocks and wherein the method further comprises a step of generating any block of the requested Web page not retrieved from the memory.

58. **(Original)** The method of claim 57, further including a step of storing a copy of any generated block in the memory.

59. **(Original)** The method of claim 57, further including the step of sending the assembled Web page over the computer network.

60. **(Original)** The method of claim 55, wherein the caching properties include at least one of a unique identifier, an expiration date, an expiration time and an invalidation rule.